Monitoreo de redes con Munin

Gunnar Eyal Wolf Iszaevich — gwolf@gwolf.org
http://ru.iiec.unam.mx/2541/

Instituto de Investigaciones Económicas • UNAM

Seminario *AdminUNAM 2014*
20 de junio 2014, DGTIC, UNAM

¿Qué hace un administrador de red/sistemas/base de datos?

- Asegurar la disponibilidad de los servicios
- Asegurar la confianza en la información provista/almacenada
- Conocer en todo momento el estado de sus equipos, para poder anticiparse a los problemas
- ...Y, claro, mucho más

Munin es una infraestructura genérica de monitoreo histórico y graficación de servicios/recursos

¿Monitoreo histórico?

Una de las tareas más importantes que, como administradores de sistemas o de redes, tenemos la responsabilidad de realizar periódicamente, para encontrar tendencias y anticiparse a los problemas, es el

Monitoreo de recursos

¿Qué significa? ¿Por qué tengo que hacerlo? ¿De qué me sirve? ¿Y cómo puedo hacerlo atractivo, incluso divertido?

El verdadero valor de las gráficas

- Las gráficas históricas son una herramienta fundamental en el arsenal de un administrador de sistemas
- El que conserven memoria histórica es fundamental para analizar tendencias y encontrar comportamientos a largo plazo
- Entre menos tengamos que configurar/batallar para lograr una primer imagen satisfactoria del sistema de monitoreo, más vamos a animarnos a utilizarlo
- Entre más podamos adecuar la información que nos es presentada a las necesidades específicas de nuestro entorno, más útil nos va a ser el sistema que elijamos
- **Nos ayudan muchísimo a aprender**, a comprender el funcionamiento y relaciones dentro de nuestro sistema

¿Qué es *Munin*?

- Una herramienta orientada a trabajo en red, integrando múltiples sistemas bajo una sóla interfaz
- Monitoreo histórico de recursos que puede ayudar –a través de la graficación– a analizar tendencias en su uso
- Ofrece una instalación muy simple, *plug-and-play*
- El desarrollo de agentes monitores para necesidades adicionales que tengamos es muy simple
- La interfaz pública no requiere de privilegio alguno de ejecución de código (produce HTML estático)
- Está construído empleando la base de datos *circular* RRDtool

Munin y Sugin

Hugin y Munin en los hombros de Odín;

manuscrito islandés, s. XVIII

Suginn ok Muninn fljúga hverjan dag Jörmungrund yfir óumk ek of Sugin, at hann aftr né komi-t, thó sjámk meir of Munin.

Hugin y Munin vuelan todos los días alrededor del mundo;
Temo menos por Hugin de que no regrese, aún más temo por Munin

Edda poética - Grímnismál, estrofas 19 y 20

Munin y Sugin

Representación de Munin y Hugin

Odín en el detalle de un casco,

En la mitología nórdica, Munin y Hugin son los cuervos del dios Odín. Vuelan a través del mundo, y relatan a Odín, susurrando a sus oídos, lo que han visto, todas las noticias.

Recuerdan todo.

Hugin es el *pensamiento* y Munin es la *memoria*.

Es debido a estos cuervos que el apelativo «Hrafnaguð» (dios cuervo) se utilizaba para referirse a Odín.

¿Qué **no es** Munin?

Munin **no es perfecto** ni es para todas las situaciones y configuraciones

- Munin recibe la información *sin autenticación* y *en texto plano* sobre la red. Si hay información sensible o confidencial, no es adecuado
- Hace monitoreo *periódico*, cada cinco minutos. Sirve para recolectar datos estadísticos, *no como herramienta de alertamiento* (puede servir, pero no es su función natural ni óptima)

Servidor, plugins y cliente

Munin se divide en tres componentes principales:

Servidor Un demonio que corre en todas las máquinas monitoreadas, por default en el puerto 4949. Su función es configurar y llamar a los plugins. Cuando hablamos de `munin-node`, nos referimos al servidor.

Plugins Cada uno de los agentes de recolección de datos que son invocados por `munin-node`. Dan la información que monitorean, y son también capaces de *describir su función y configuración*

Cliente Proceso que corre periódicamente (normalmente cada 5 minutos) desde un nodo central, interrogando a cada uno de los servidores `munin-node`, y generando las páginas Web con los resultados

Instalación de Munin

Puedo asumir que utilizan software de calidad, que no les gusta complicarse la vida...
¿Verdad?

Instalación de Munin

Puedo asumir que utilizan software de calidad, que no les gusta complicarse la vida...
¿Verdad?

Instalando Munin en Debian

```
gwolf@malenkaya[1] $ su -
Password:
root@malenkaya[1] # apt-get install munin munin-node
(...)
Do you want to continue? [Y/n/?] y
(...)
```

Y ya.

Instalado, configurado y funcionando.

Explicando la magia: munin-node-config

Un buen mago jamás revela sus secretos.

Explicando la magia: munin-node-config

Un buen mago jamás revela sus secretos.
Afortunadamente, soy administrador de sistemas, no mago.

munin-node-configure

Llamado sin opciones, nos desglosa la configuración actual:

```
0 gwolf@malenkaya[1]~$ /usr/sbin/munin-node-configure
Plugin                  | Used | Extra information
------                  | ---- | -----------------
acpi                    | yes  |
apache_accesses         | no   |
apache_processes        | no   |
apache_volume           | no   |
apt                     | no   |
apt_all                 | no   |
courier_mta_mailqueue   | no   |
courier_mta_mailstats   | no   |
courier_mta_mailvolume  | no   |
                        (...)
hddtemp_smartctl        | no   |
if_                     | yes  | eth1 eth2
if_err_                 | no   |
interrupts              | yes  |
```

Explicando la magia: munin-node-config

Indicando -suggest, nos *sugiere* qué plugins activar y con qué configuraciones lanzarlos

munin-node-configure -suggest

```
0 gwolf@malenkaya[2]~$ /usr/sbin/munin-node-configure --suggest
Plugin                  | Used | Suggestions
------                  | ---- | -----------
apache_accesses         | no   | [no apache server-status or ExtendedStatus missing on ports 80]
apache_processes        | no   | [no apache server-status on ports 80]
apache_volume           | no   | [no apache server-status or ExtendedStatus missing on ports 80]
courier_mta_mailqueue   | no   | [spooldir not found]
courier_mta_mailstats   | no   | [could not find executable]
courier_mta_mailvolume  | no   | [could not find executable]
cupsys_pages            | no   | [logfile not readable]
exim_mailqueue          | no   | [exim not found]
exim_mailstats          | no   |
hddtemp_smartctl        | no   | [smartctl not found]
if_                     | yes  | -eth2
if_err_                 | no   | yes +eth1
```

Claro, al momento de instalación, el sistema es interrogado por este script — Y obtenemos una configuración por omisión muy completa.

Configurando Munin-node (servidor) más allá de los defaults

¿Y si quiero afinar lo que me es monitoreado? ¿Agregar, quitar agentes? ¿Especificar parámetros?
Los parámetros base de cada servidor `munin-node` son configurados en /etc/munin/munin-node.conf:

```
log_level 4
log_file /var/log/munin/munin-node.log
port 4949
pid_file /var/run/munin/munin-node.pid
background 1
setseid 1

# Which port to bind to;
host 127.0.0.1
user root
group root
setsid yes
```

Para un sólo nodo, la configuración default es típicamente adecuada.

Configurando Munin-node (servidor) más allá de los defaults

Para determinar los plugins a activar, basta ligarlos o eliminarlos de /etc/munin/plugins:

```
lrwxrwxrwx 1 root root 29 2008-03-22 16:06 acpi -> /usr/share/munin/plugins/acpi
lrwxrwxrwx 1 root root 28 2008-03-22 16:06 cpu -> /usr/share/munin/plugins/cpu
lrwxrwxrwx 1 root root 27 2008-03-22 16:06 df -> /usr/share/munin/plugins/df
lrwxrwxrwx 1 root root 33 2008-03-22 16:06 df_inode -> /usr/share/munin/plugins/df_inode
lrwxrwxrwx 1 root root 32 2008-03-22 16:06 entropy -> /usr/share/munin/plugins/entropy
lrwxrwxrwx 1 root root 30 2008-03-22 16:06 forks -> /usr/share/munin/plugins/forks
lrwxrwxrwx 1 root root 28 2008-03-22 16:06 if_eth1 -> /usr/share/munin/plugins/if_
lrwxrwxrwx 1 root root 28 2008-04-05 15:21 if_eth2 -> /usr/share/munin/plugins/if_
(...)
```

Tomen nota de los últimos dos — Los plugins cuyo nombre termina en _ son *mágicos*: El nombre que liga hacia el plugin es tomado como argumento

Configurando Munin-node (servidor) más allá de los defaults

- Algunos plugins pueden recibir, además, parámetros o configuraciones adicionales.
- Estos los configuramos a través de archivos en `/etc/munin/plugin-conf.d/`, en bloques con el estilo general/tradicional `.ini`
- ...En una instalación básica, rara vez hace falta siquiera tocar estos archivos

Configurando Munin (cliente) más allá de los defaults

¿Y cómo monitoreo a varios hosts?
La configuración del cliente esta en /etc/munin/munin.conf; la sección relevante default dice:

```
[localhost.localdomain]
    address 127.0.0.1
    use_node_name yes
```

Configurando Munin (cliente) más allá de los defaults

Para monitorear a varios hosts, es casi igual:

```
[webserver.localdomain]
    address 127.0.0.1
    use_node_name yes

[database.localdomain]
    address 192.168.100.150
    use_node_name yes
```

Hay varias opciones más disponibles para la generación de páginas de gráficas (p.ej. totalizaciones, ordenamiento interno...)

Lógica de operación de un plugin

- Cada plugin es ejecutado por munin-node como un script cualquiera.
- El plugin corre con los privilegios de usuario definidos en su entrada en `/etc/munin/plugin-conf.d/`
- Su modo de operación lo determina el único parámetro

`autoconf`	¿Es capaz de autoconfigurarse en base al entorno? (`yes` / `no` y estado de salida)
`suggest`	¿En qué casos se *autosugerirá* al cliente Munin? (por estado de salida)
`config`	¿Qué parámetros graficará? (Descripción de la gráfica, etiquetas de los ejes y las variables, e indicaciones de formato a RRDtool
`fetch`	Entrega los resultados del monitoreo en cuestión. Es el comportamiento default.

El protocolo básico de Munin

El servidor de munin implementa un protocolo muy sencillo:

list	Muestra los plugins disponibles en este host
nodes	Nodos que este host reporta (p.ej. sobre SNMP)
config *plugin*	Descripción y configuración del plugin especificado
fetch *plugin*	Recupera los valores actuales del plugin solicitado
quit	Finaliza la sesión de monitoreo

Una conversación ejemplo con Munin-node (1)

```
$ nc localhost 4949
# munin node at webserver.localdomain
help
# Unknown command. Try list, nodes, config, fetch, version or quit
list
open_inodes entropy irqstats mysql_slowqueries processes acpi
mysql_threads df netstat interrupts swap mysql_bytes if_eth2 load
df_inode cpu if_eth1 mysql_queries forks iostat open_files memory vmstat
nodes
webserver.localdomain
.
```

Una conversación ejemplo con Munin-node (1)

```
$ nc localhost 4949
# munin node at webserver.localdomain
help
# Unknown command. Try list, nodes, config, fetch, version or quit
list
open_inodes entropy irqstats mysql_slowqueries processes acpi
mysql_threads df netstat interrupts swap mysql_bytes if_eth2 load
df_inode cpu if_eth1 mysql_queries forks iostat open_files memory vmstat
nodes
webserver.localdomain
.
```

Una conversación ejemplo con Munin-node (1)

```
$ nc localhost 4949
# munin node at webserver.localdomain
help
# Unknown command. Try list, nodes, config, fetch, version or quit
list
open_inodes entropy irqstats mysql_slowqueries processes acpi
mysql_threads df netstat interrupts swap mysql_bytes if_eth2 load
df_inode cpu if_eth1 mysql_queries forks iostat open_files memory vmstat
nodes
webserver.localdomain
.
```

Una conversación ejemplo con Munin-node (1)

```
$ nc localhost 4949
# munin node at webserver.localdomain
help
# Unknown command. Try list, nodes, config, fetch, version or quit
list
open_inodes entropy irqstats mysql_slowqueries processes acpi
mysql_threads df netstat interrupts swap mysql_bytes if_eth2 load
df_inode cpu if_eth1 mysql_queries forks iostat open_files memory vmstat
nodes
webserver.localdomain
.
```

Una conversación ejemplo con Munin-node (2)

```
config entropy
graph_title Available entropy
graph_args -base 1000 -l 0
graph_vlabel entropy (bytes)
graph_scale no
graph_category system
graph_info This graph shows the amount of entropy available in the
system.
entropy.label entropy
entropy.info The number of random bytes available. This is typically
used by cryptographic applications.
.
fetch entropy
entropy.value 815
```

Una conversación ejemplo con Munin-node (2)

```
config entropy
graph_title Available entropy
graph_args -base 1000 -l 0
graph_vlabel entropy (bytes)
graph_scale no
graph_category system
graph_info This graph shows the amount of entropy available in the
system.
entropy.label entropy
entropy.info The number of random bytes available. This is typically
used by cryptographic applications.
.
fetch entropy
entropy.value 815
```

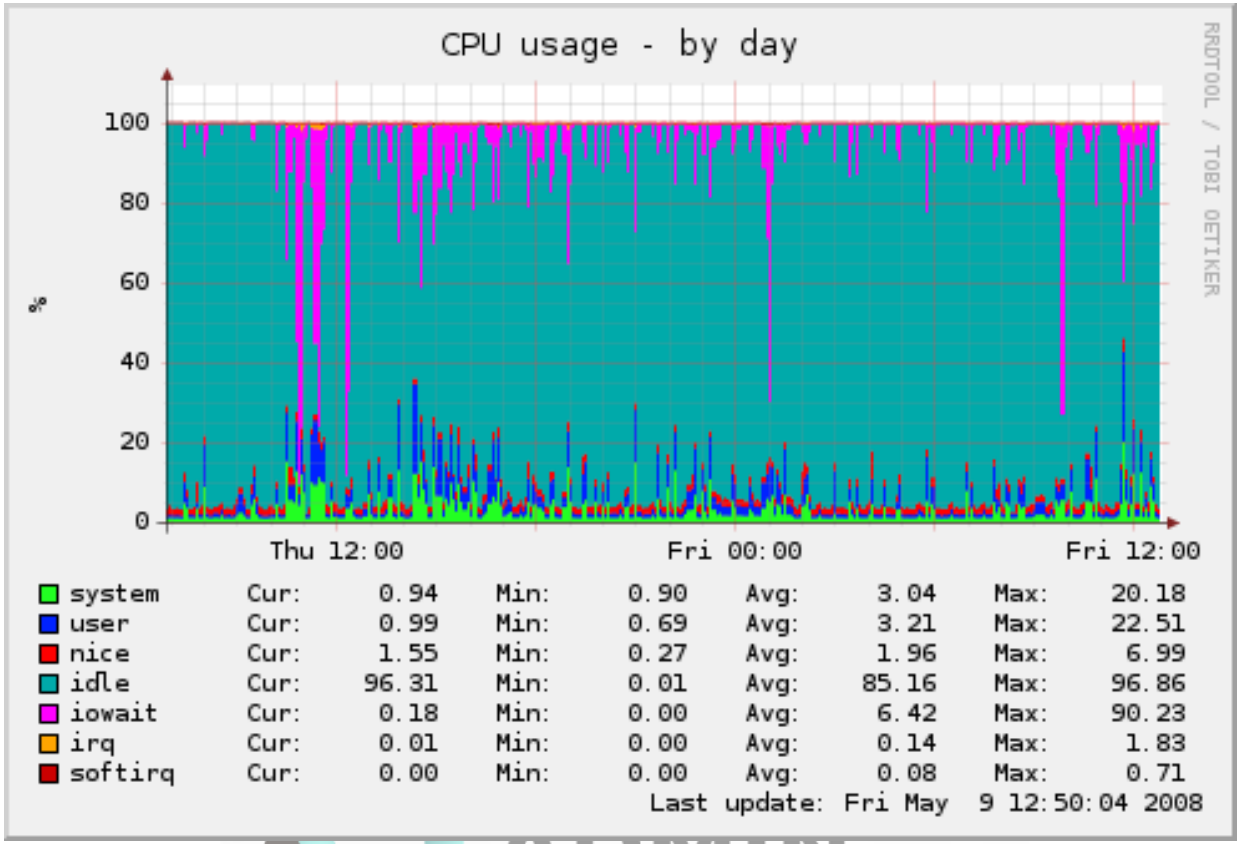
Una conversación ejemplo con Munin-node (2)

```
config entropy
graph_title Available entropy
graph_args -base 1000 -l 0
graph_vlabel entropy (bytes)
graph_scale no
graph_category system
graph_info This graph shows the amount of entropy available in the
system.
entropy.label entropy
entropy.info The number of random bytes available. This is typically
used by cryptographic applications.
.
fetch entropy
entropy.value 815
```

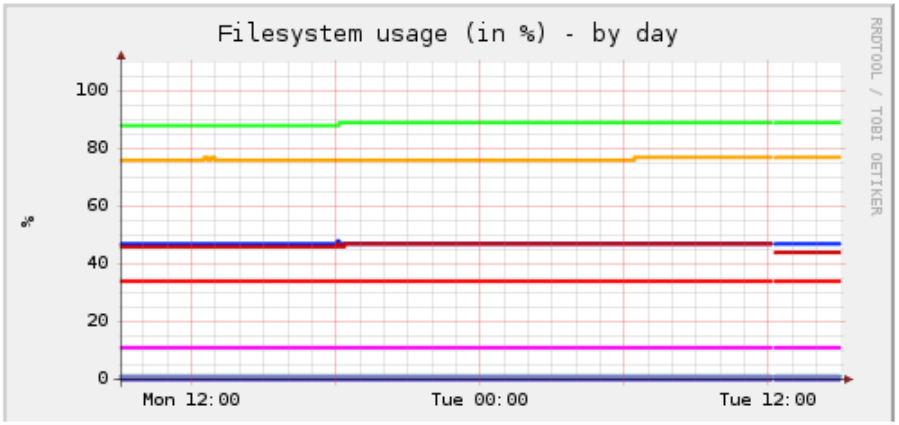
Leyendo gráficas como mecanismo de aprendizaje

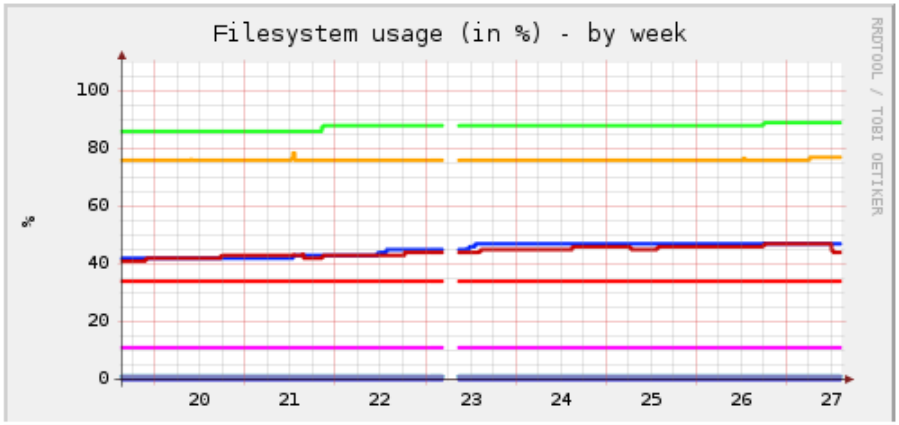
- Leer las gráficas generadas por Munin nos puede llevar a aprender y entender muchos aspectos del funcionamiento de nuestro sistema
- Es importante que revisemos todos los aspectos que llamen nuestra atención — Pueden llevarnos a sorprendentes revelaciones
- Munin viene pre-configurado para monitorear diversos aspectos del sistema — ¡Aprovechémoslo!

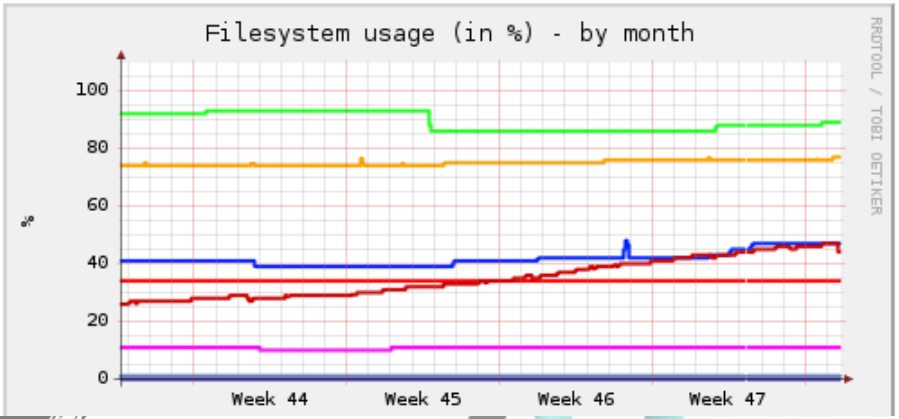
Uso de CPU

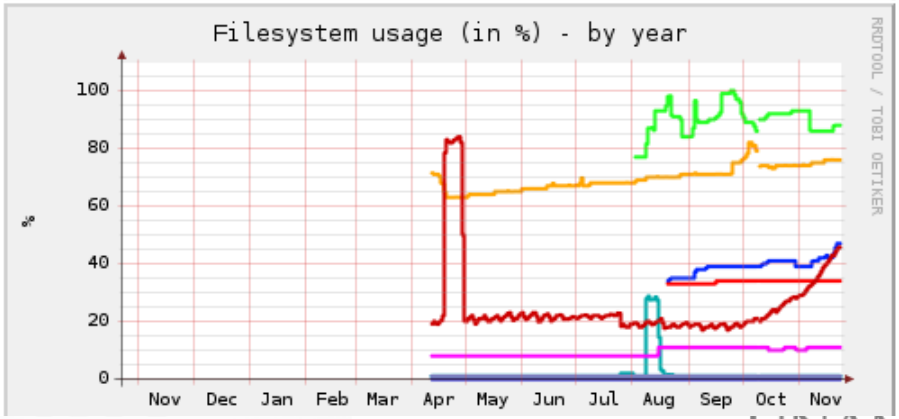

CPU usage - by day
RRDTOOL / TOBI OETIKER
%
100
80
60
40
20
0
Thu 12:00
Fri 00:00
Fri 12:00
system Cur: 0.94 Min: 0.90 Avg: 3.04 Max: 20.18
user Cur: 0.99 Min: 0.69 Avg: 3.21 Max: 22.51
nice Cur: 1.55 Min: 0.27 Avg: 1.96 Max: 6.99
idle Cur: 96.31 Min: 0.01 Avg: 85.16 Max: 96.86
iowait Cur: 0.18 Min: 0.00 Avg: 6.42 Max: 90.23
irq Cur: 0.01 Min: 0.00 Avg: 0.14 Max: 1.83
softirq Cur: 0.00 Min: 0.00 Avg: 0.08 Max: 0.71
Last update: Fri May 9 12:50:04 2008

Uso (anual) del sistema de archivos

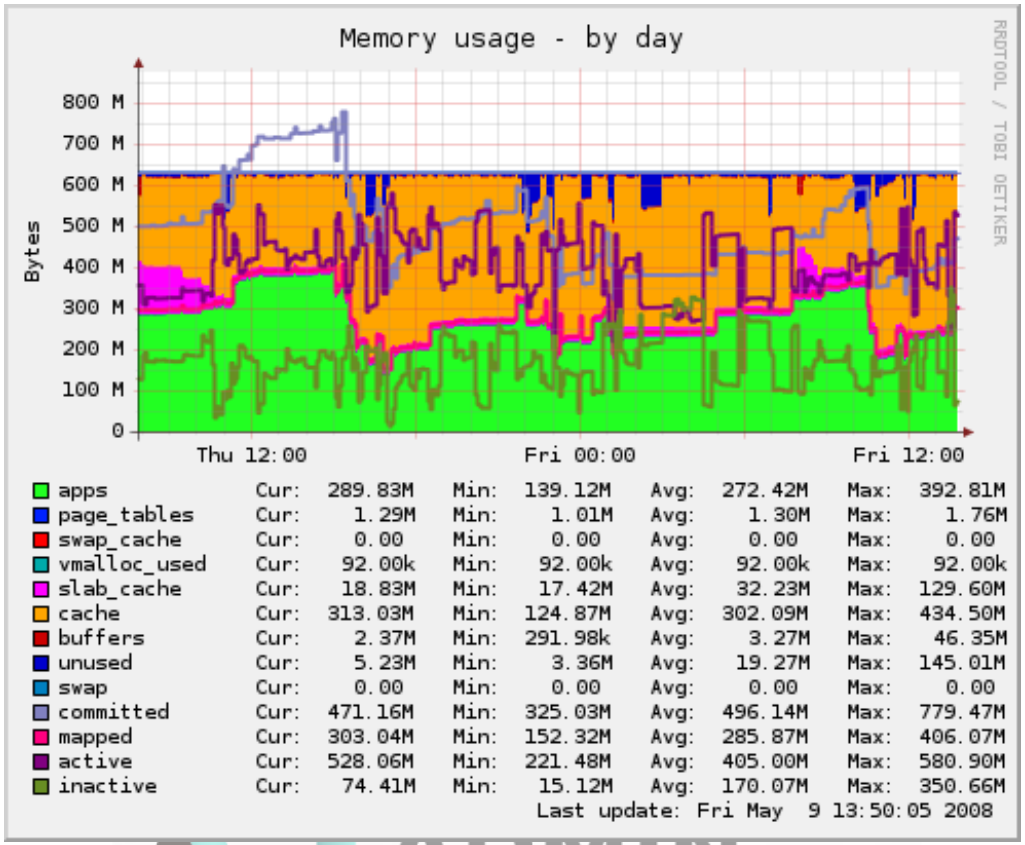




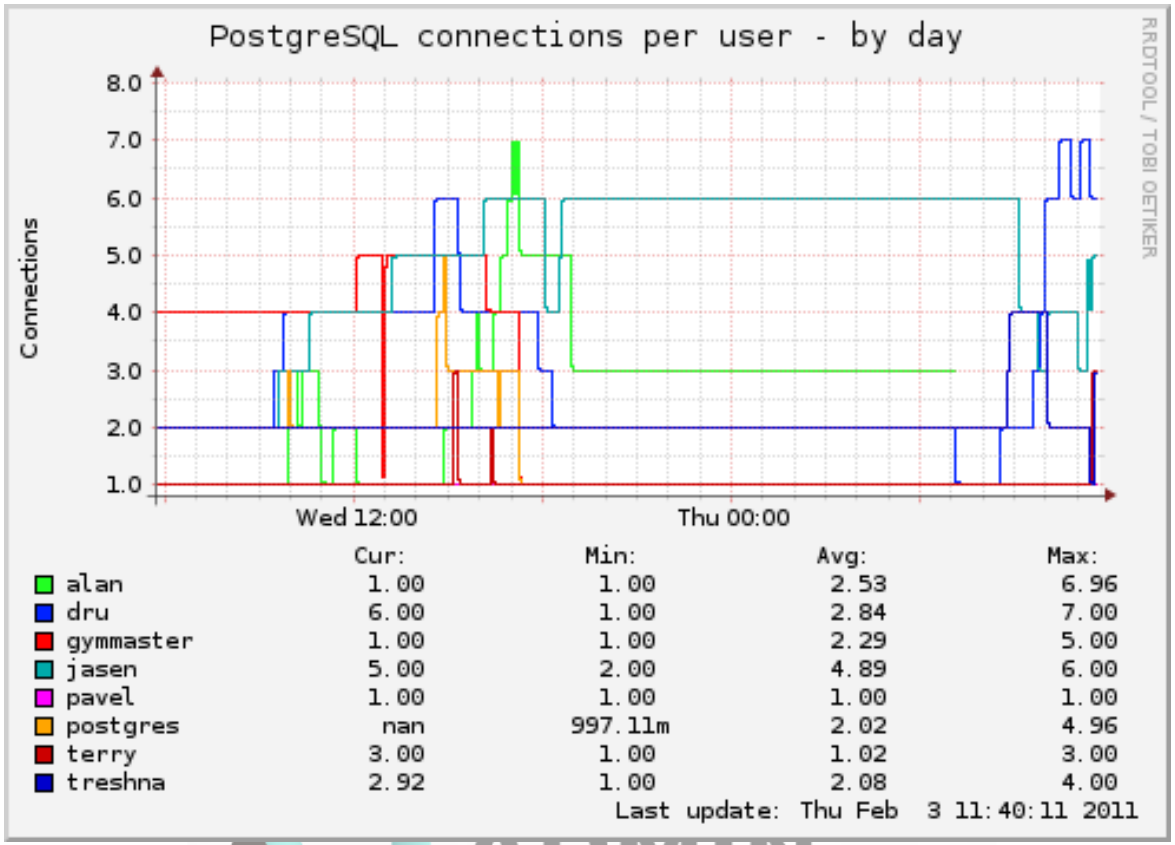




Uso de memoria



Conexiones a PostgreSQL por usuario



¡El cielo es el límite!

- Probablemente la mayor fuerza de Munin es lo fácil que resulta implementar y desplegar nuestros propios plugins
- Los plugins pueden implementarse en cualquier lenguaje, basta con ofrecer una interfaz muy simple
- Cualquier cosa cuantificable es monitoreable
- Para realmente aprovecharlo en realidad, nos conviene aprender las sutilezas de RRDtool

¿Y cómo obtengo los datos?

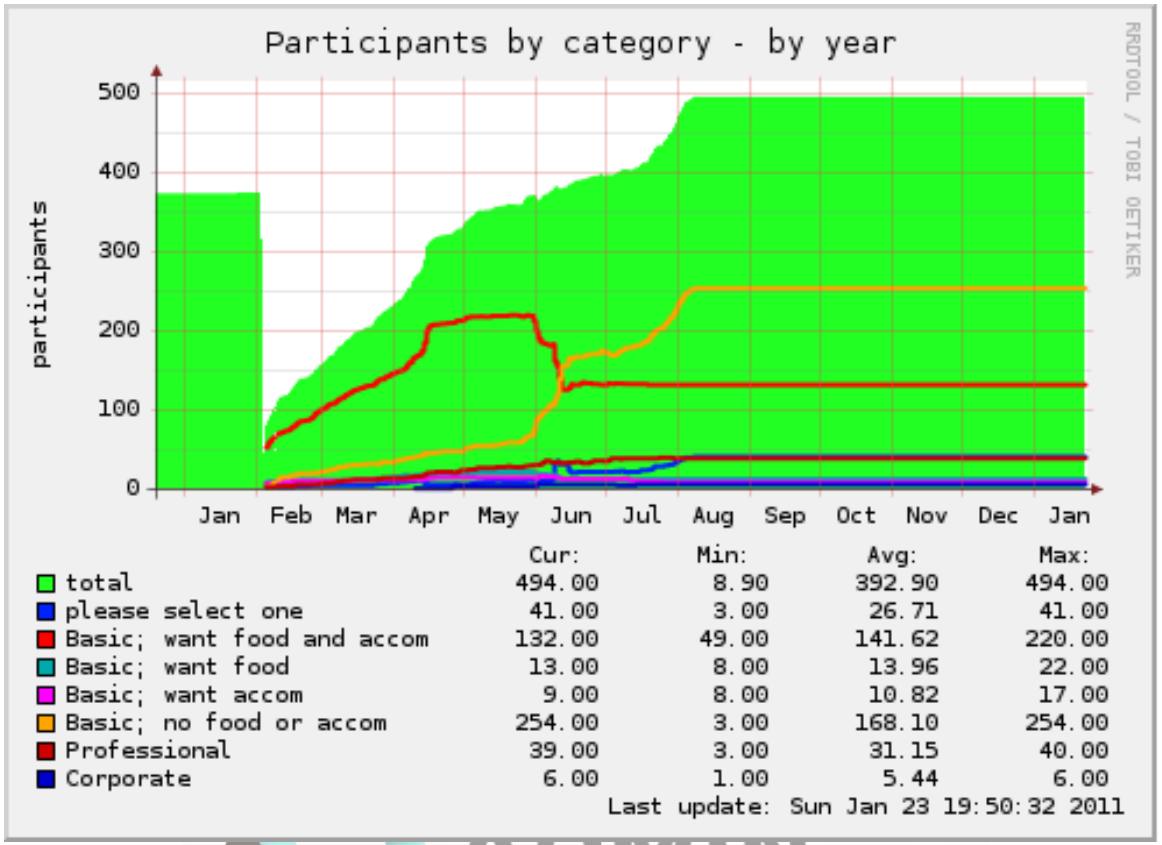
- Cualquier programa que implemente el protocolo de Munin puede ser usado como plugin
- El módulo de Perl `Munin::Plugin::Pgsql` hace casi trivial crear plugins implementando una consulta arbitraria
- Podemos graficar la *salud* de PostgreSQL, los datos mismos almacenados en nuestra base de datos, o cualquier agregado que se nos ocurra

Utilizando Munin::Plugin::Pgsql

Ejemplo: Número de conexiones por usuario

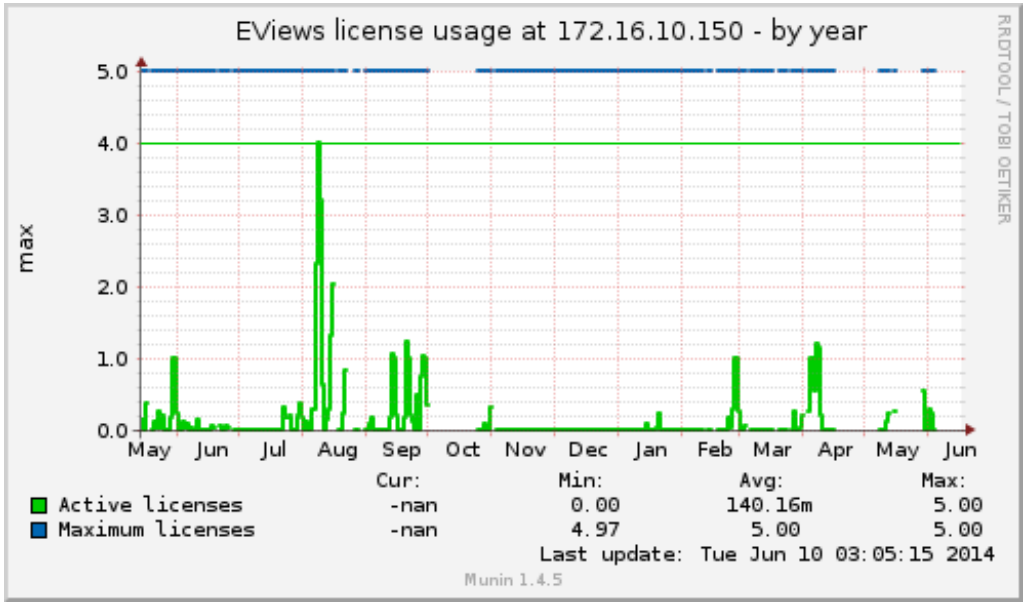
```
#!/usr/bin/perl
use strict;
use warnings;
use Munin::Plugin::Pgsql;
my $pg = Munin::Plugin::Pgsql->new(
    title  => 'PostgreSQL connections per user',
    info   => 'Number of connections per user',
    vlabel => 'Connections',
    basequery => "SELECT usename,count(*) FROM pg_stat_activity WHERE " .
            "procpid != pg_backend_pid() GROUP BY usename ORDER BY 1",
    configquery => "SELECT DISTINCT usename,usename FROM " .
                   "pg_stat_activity ORDER BY 1");
$pg->Process();
```

Número de participantes en un congreso, por categoría



Graficación de servicios que *no cooperan*

Incluso los servicios que están hechos para *no ser monitoreados* pueden incluirse con un poco de ingenio



¿Costo? Un programita bastante simple de 200 líneas en Ruby.
¿Ahorro? Del órden de 700 dólares anuales.

Cosas a tomar en cuenta con nuestros plugins

- ¿Con qué privilegios va a correr? ¿Puedo *obligar* a que el script no sea llamado como root?
- ¿Qué tiempo de ejecución tiene cada plugin? Recuerden que voy a tener una invocación de *todos* mis plugins cada cinco minutos. ¿Puedo separarlo en hilos o procesos?
- ¿Qué tan sensibles son estos datos? Recuerden que Munin **no implementa autenticación ni cifado** — Pueden configurarlo p.ej. sobre interfaces ligadas a una VPN.

Concluyendo

Munin es...

- Bonito
- Divertido
- Una herramienta para monitorear nuestros sistemas
- Un gran recurso para *aprender* acerca de nuestros sistemas
- Una maravillosa herramienta para apantallar al jefe
- Una *excelente* manera de perder el tiempo

¿Dudas?

¡Gracias!

Gunnar Wolf — gwolf@gwolf.org
Instituto de Investigaciones Económicas • UNAM

http://ru.iiec.unam.mx/2541/